



Уральский
федеральный
университет

Параллельные вычисления

MapReduce и Hadoop

Созыкин Андрей Владимирович

К.Т.Н.

зав.кафедрой высокопроизводительных компьютерных технологий

Big Data

- Big Data – задачи обработки больших объемов данных:
 - Терабайты и петабайты
 - Неструктурированные данные
 - Высокая скорость обработки – невозможно сделать традиционными подходами
- Особенности Big Data
 - «Простые» операции с данными, не занимающие много времени
 - Основное время тратится на передачу данных для обработки
- Перемещение вычислений к данным

MapReduce

- MapReduce – доминирующая технология обработки больших объемов данных
- MapReduce включает:
 - Модель программирования
 - Среда выполнения
 - Программная реализация

История создания MapReduce

- Технологию MapReduce придумали в Google для системы поиска в Интернет:
 - Цель – хранить и обрабатывать большие объемы данных на обычных компьютерах, объединенных сетью
 - Jeffrey Dean, Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters
 - Sanjay Ghemawat, Howard Gobioff, Shun-Tak Leung. The Google File System
- Задачи поисковой системы:
 - Сбор содержимого Web (crawling)
 - Построение инвертированного индекса
 - Ранжирование документов для ответа на запросы

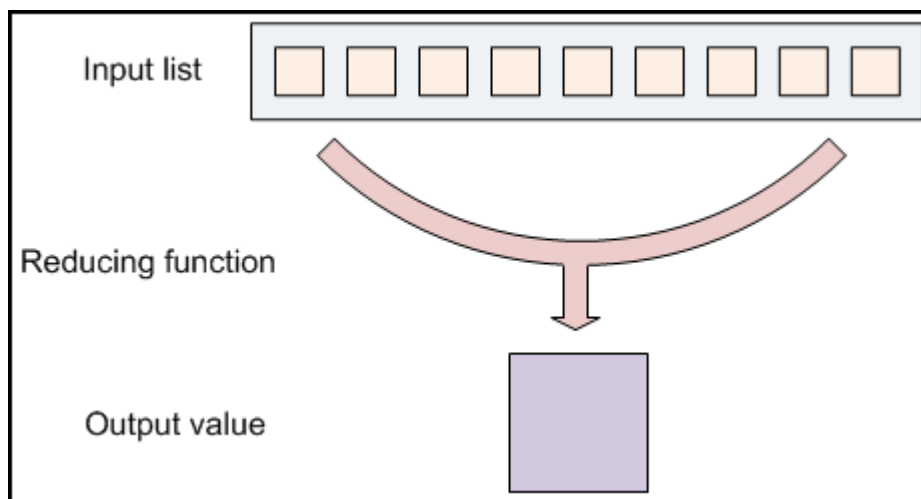
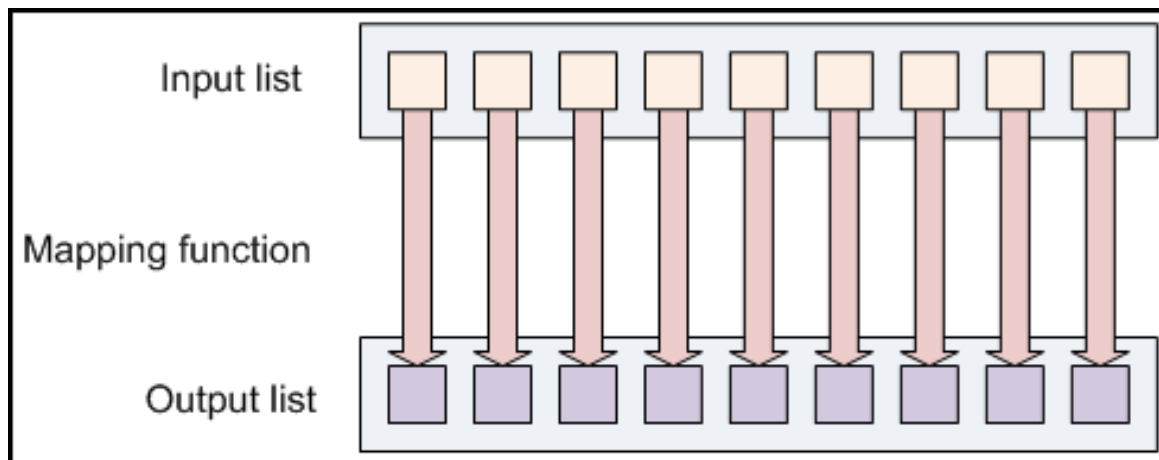
Особенности MapReduce

- Автоматизация распараллеливания
- Распределенная файловая система для хранения данных на внутренних дисках серверов кластера
- Ориентация на потоковую обработку больших объемов данных
- Защита от сбоев оборудования
 - Автоматический перезапуск задач на других узлах
- Обработка данных там, где они хранятся
 - Перемещение вычислений к данным
- Один алгоритм обработки данных - MapReduce

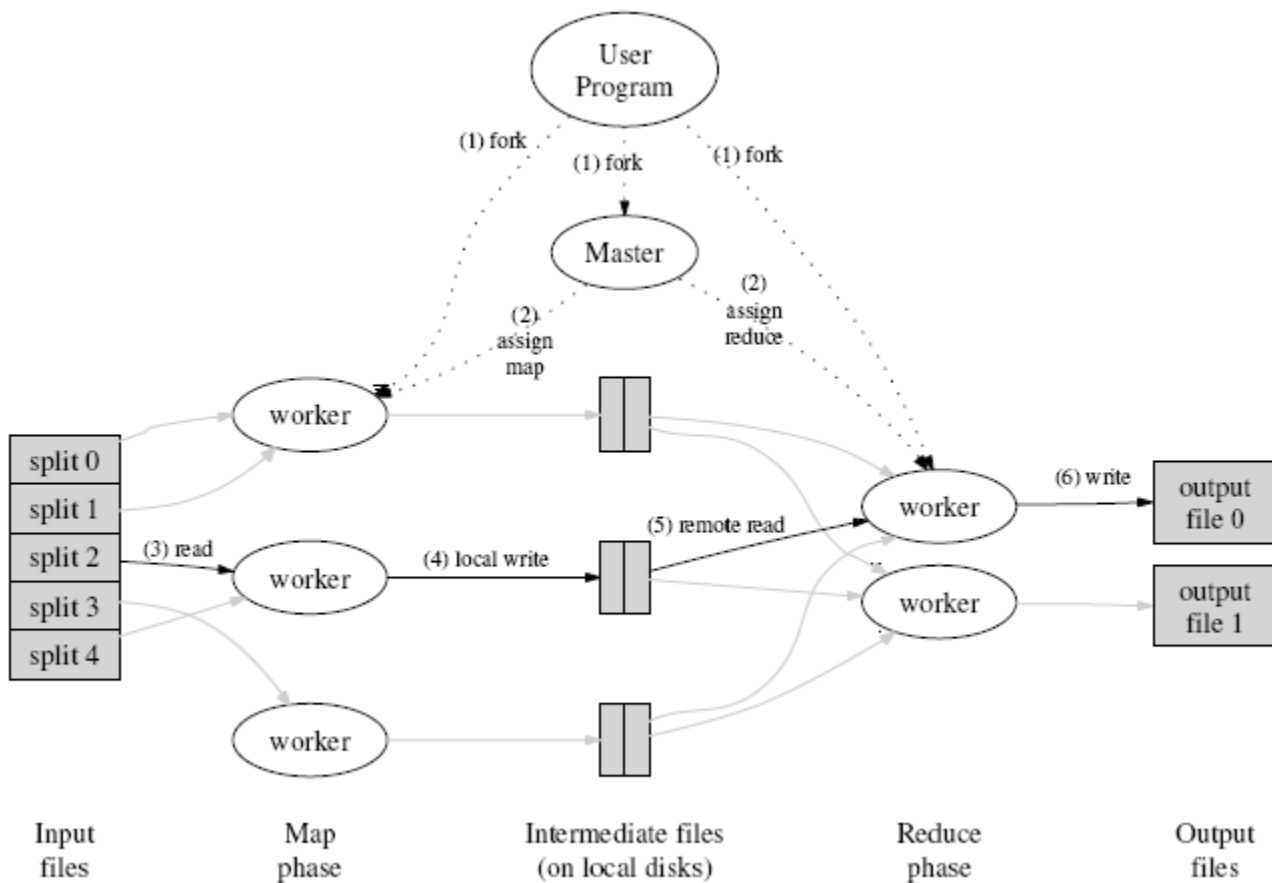
MapReduce

- Цель MapReduce – разделить логику приложения и организацию распределенного взаимодействия:
 - Программист реализует только логику приложения
 - Распределенная работа в кластере обеспечивается автоматически
- MapReduce работает с данными как с парами Ключ:Значение:
 - смещение в файле: текст
 - идентификатор пользователя: профиль
 - пользователь: список друзей
 - временная метка: событие в журнале

Функции Map и Reduce



Архитектура системы MapReduce



Пример MapReduce: WordCount

- Задача: посчитать, сколько раз слово встречается в файле
 - Количество обращений к Web-страницам
 - Количество просмотра видео или прослушивания песни
- Исходные данные:
 - Текстовые файлы
 - Каждый файл делится на пары: Смещение:Текст
- Пример:
 - Цель MapReduce – разделить логику приложения и организацию распределенного взаимодействия. Программист реализует только логику приложения

WordCount: функция Map

- Исходные данные:
 - Цель MapReduce – разделить логику приложения и организацию распределенного взаимодействия. Программист реализует только логику приложения
- Результаты обработки:
 - <цель, 1>, <mapreduce,1>, <разделить, 1>, <логику,1>, <приложения, 1>, <и, 1>, <организацию, 1>, <распределенного, 1>, <взаимодействия, 1>, <программист, 1>, <реализует, 1>, <только,1>, <логику, 1>, <приложения, 1>
- Сортировка и группировка по ключу:
 - <mapreduce,1>, <взаимодействия, 1>, <и, 1>, <логику,1>, <логику, 1>, <организацию, 1>, <приложения, 1>, <приложения, 1>, <программист, 1>, <разделить, 1>, <распределенного, 1>, <реализует, 1>, <только,1>, <цель, 1>.

WordCount: функция Reduce

- Пары с одинаковыми ключами передаются в одну функцию Reduce:

<mapreduce,1> → <mapreduce,1>

<взаимодействия, 1> → <взаимодействия, 1>

<и, 1> → <и, 1>

<логику,1>, <логику, 1> → <логику, 2>

<организацию, 1> → <организацию, 1>

<приложения, 1>, <приложения, 1> → <приложения, 2>

<программист, 1> → <программист, 1>

<разделить, 1> → <разделить, 1>

<распределенного, 1> → <распределенного, 1>

<реализует, 1> → <реализует, 1>

<только,1> → <только,1>

<цель, 1> → <цель, 1>

Модель MapReduce

- MapReduce – странный подход к решению задачи WordCount
 - Есть более простые и интуитивно понятные решения
- Достоинства MapReduce:
 - Возможность автоматического распараллеливания – функции Map и Reduce могут обрабатывать элементы списка параллельно не зависимо друг от друга
 - Масштабируемость – данные могут размещаться на разных серверах и обрабатываться также на разных серверах
 - Отказоустойчивость – при выходе из строя сервера функции Map или Reduce запускаются на другом сервере
- Недостатки MapReduce:
 - Фиксированный алгоритм обработки данных
 - Высокие накладные расходы на распараллеливание

Модель MapReduce: другие примеры

- Поиск в тексте
 - map: (docid, content) \rightarrow [(docid, line)]
 - reduce: нет
- Обращение Web-графа:
 - map: (docid, content) \rightarrow [(url, docid)]
 - reduce: (url, [docid]) \rightarrow (url, [docid])
- Анализ посещаемости сайта:
 - map: (logid, log) \rightarrow [(url, visit_count)]
 - reduce: (url, [visit_count]) \rightarrow (url, total_count)
- Вычисление векторов ключевых слов по сайтам:
 - map: (docid, <url, content>) \rightarrow (hostname, doc_term_vector)
 - reduce: (hostname, [doc_term_vector]) \rightarrow (hostname, host_term_vector)

Реализации MapReduce

- Оригинальная реализация Google
 - Основана на C++
 - Не распространяется
- Apache Hadoop:
 - Наиболее популярная реализация MapReduce с открытыми исходными кодами
 - Язык Java
- Другие реализации

Кто использует Hadoop

- Кто использует Hadoop:

facebook

twitter

LinkedIn

YAHOO!

last.fm

amazon.com

The New York Times

eBay

- Самый большой кластер Hadoop в Yahoo!:
 - 4500 серверов
 - Используется для поисковой системы и подбора рекламных объявлений

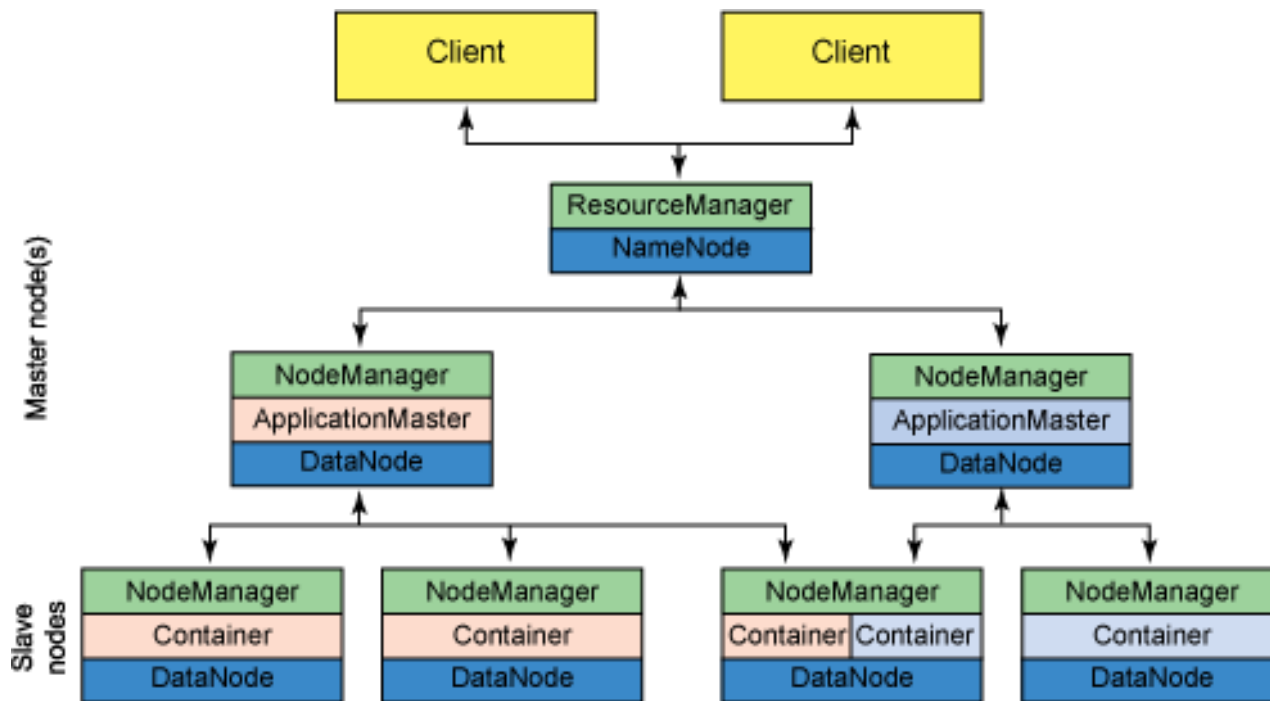
Задачи реализации MapReduce

- Декомпозиция на параллельные подзадачи (map- и reduce-задачи)
- Запуск рабочих процессов
- Распределение задач по рабочим процессам и балансировка нагрузки
- Передача данных рабочим процессам (требуется минимизировать)
- Синхронизация и передача данных между рабочими процессами
- Обработка отказов рабочих процессов

Основные технологии Hadoop

- HDFS (Hadoop Distributed File System) – хранение данных
- MapReduce – обработка данных

Архитектура Hadoop v2



<http://www.ibm.com/developerworks/library/bd-hadoop yarn/>

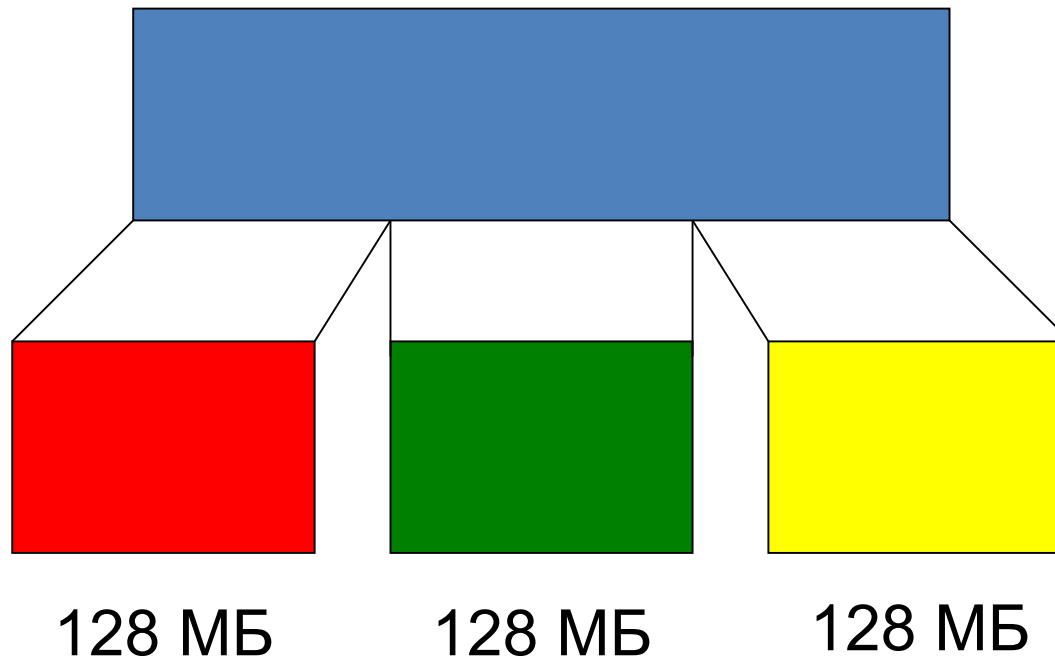
HDFS

Файл



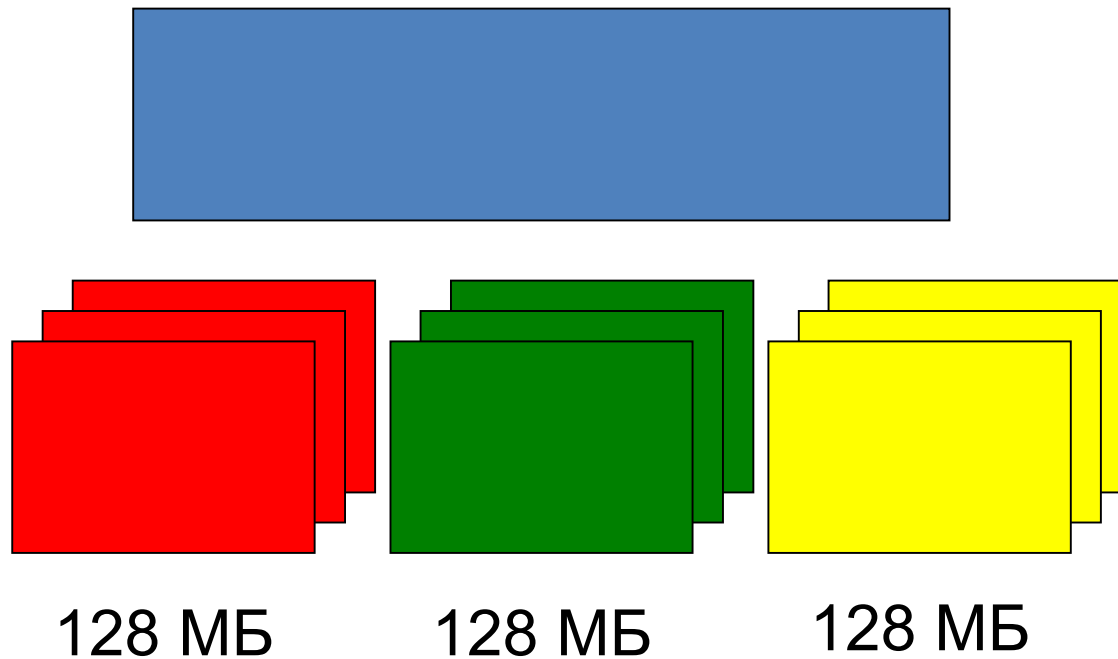
HDFS

Файл

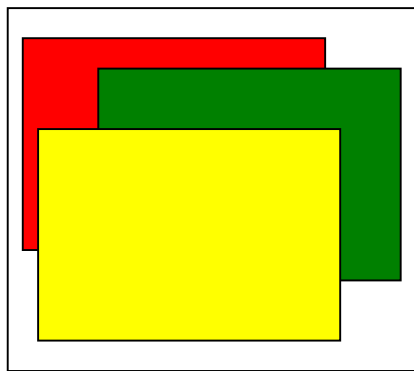


HDFS

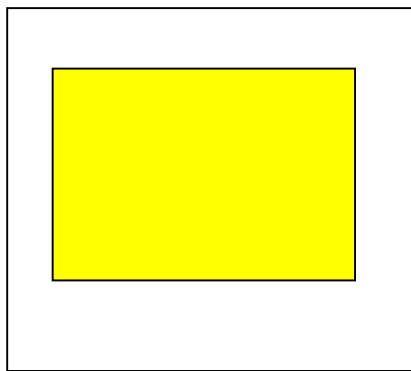
Файл



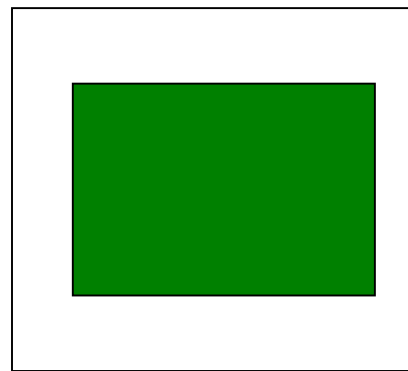
HDFS



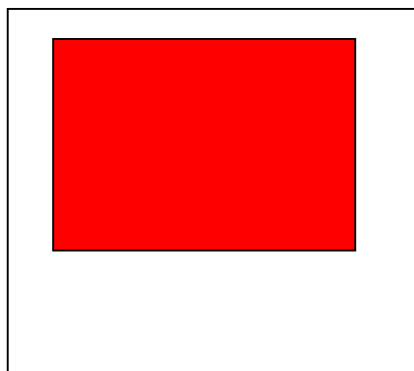
Data Node 1



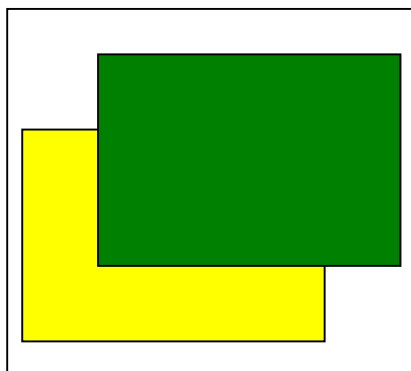
Data Node 2



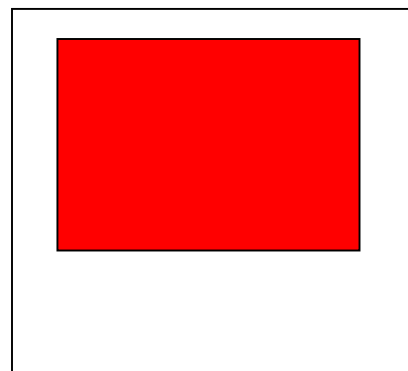
Data Node 3






Data Node 4



Data Node 5



Data Node 6

Name	Node
	1, 4, 6
	1, 3, 5
	1, 2, 5

Работа с HDFS

- Блоки файлов в HDFS распределены по разным серверам:
 - Нельзя смонтировать HDFS
 - Не работают стандартные команды ls, cp, mv и т.п.
- Необходимо использовать специальную команду:
 - `$ hdfs fs -cmd`

- Примеры:

```
$ hdfs fs -ls
```

```
Found 3 items
```

```
-rw-r--r--    1 hadoop supergroup    0 2011-06-22 13:58 /user/hadoop/file1  
-rw-r--r--    1 hadoop supergroup    0 2011-06-22 13:58 /user/hadoop/file2  
-rw-r--r--    1 hadoop supergroup    0 2011-06-22 13:58 /user/hadoop/file3
```

```
$ hdfs fs -put /tmp/file4
```

```
$ hdfs fs -cat file4
```

```
Hello, world!
```

Особенности HDFS

- HDFS – специализированная файловая система, оптимизированная для параллельной потоковой работы с большими файлами
 - Подходит не для всех задач!
- Модель Write Once Read Many:
 - Нельзя изменять файл, можно только добавлять в конец
- Большой размер блока:
 - По-молчанию 64 МБ (часто 128 или 256 МБ)
 - Не эффективен произвольный доступ (базы данных и т.п.)

MapReduce в Hadoop

- Реализация обработки данных по технологии MapReduce в Hadoop
- Язык программирования Java
 - Можно использовать другие языки с помощью технологии Streaming
- Основные компоненты MapReduce программы:
 - Функция Map
 - Функция Reduce
 - Драйвер

WordCount: Mapper

```
public class WordCountMapper
    extends Mapper<LongWritable, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, Context
        context) throws IOException, InterruptedException {
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            context.write(word, one);
        }
    }
}
```

WordCount: Reducer

```
public class WordCountReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterable<IntWritable> values,
Context context) throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
}
```

WordCount: Driver

```
public class WordCount extends Configured implements Tool {
    public static void main(String[] args) throws Exception {
        int exitCode = ToolRunner.run(new WordCount(), args);
        System.exit(exitCode);
    }

    public int run(String[] args) throws Exception {
        Job job = Job.getInstance(super.getConf(), "WordCount");
        job.setJarByClass(getClass());
        job.setMapperClass(WordCountMapper.class);
        job.setReducerClass(WordCountReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        return job.waitForCompletion(true) ? 0 : 1;
    }
}
```

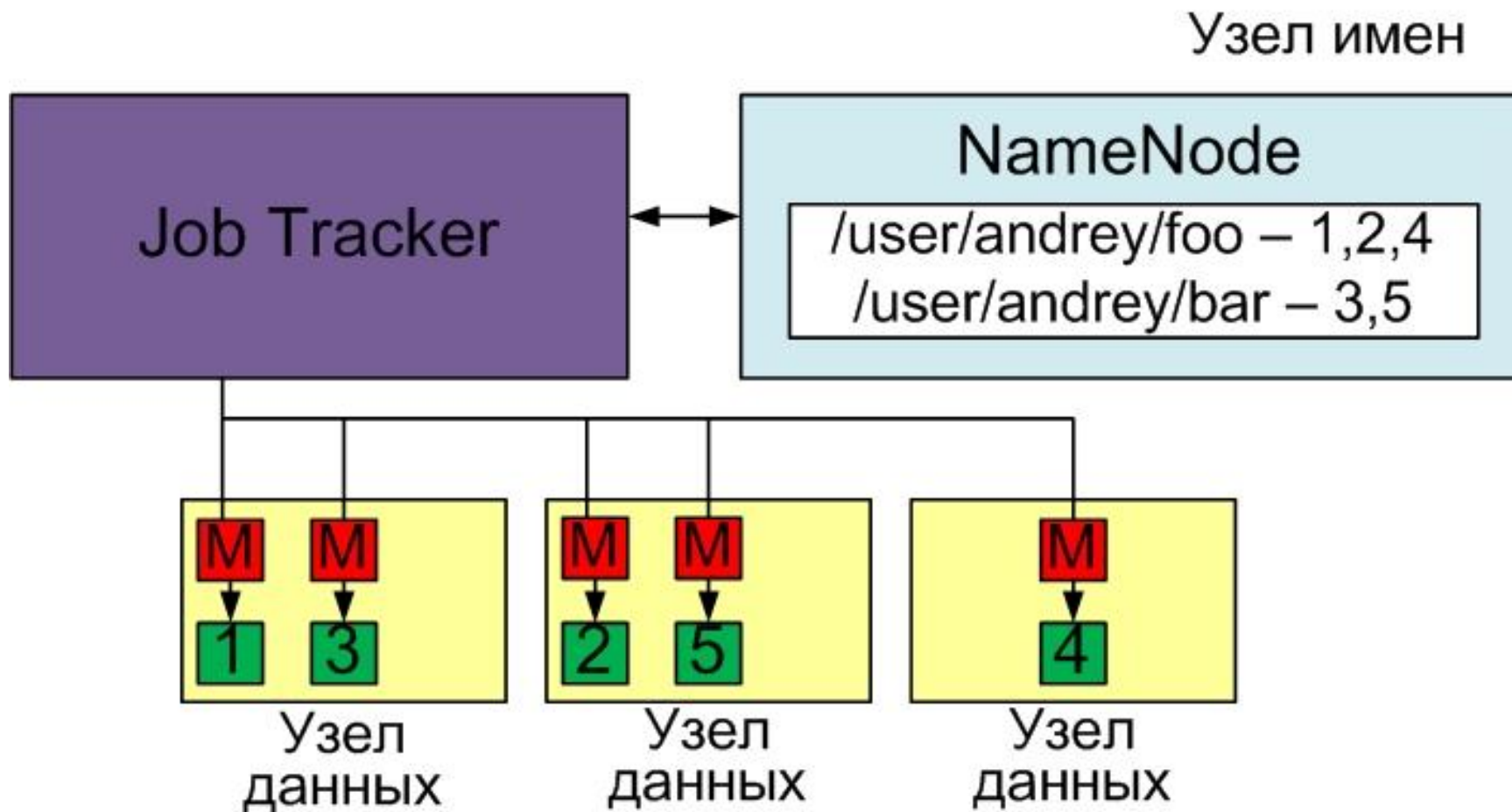
Разработка и запуск задачи MapReduce

- Реализовать Mapper, Reducer и Driver
- Скомпилировать их и упаковать в Jar-архив
- Скопировать данные для обработки в HDFS
- Запустить MapReduce задачу
- Обработанные данные будут записаны в HDFS

Пример запуска задачи Hadoop

- `$ hadoop jar wordcount.jar WordCount input output`
 - `wordcount.jar` – Имя архива с разработанной программой (Mapper, Reducer и Driver)
 - `WordCount` – Имя класса в архиве для запуска (Driver)
 - `input` – Каталог входных данных (в HDFS)
 - `output` – Каталог выходных данных (в HDFS)

Перемещение вычислений к данным



Экосистема Hadoop

- MapReduce – мощная модель программирования, но низкоуровневая
 - Реализация практически полезных алгоритмов требует высоких трудозатрат
- Hadoop сложен в установке и администрировании
- На основе Hadoop сложилась экосистема:
 - Программные продукты для решения различных прикладных задач, использующие Hadoop для масштабирования
 - Дистрибутивы Hadoop
 - Облачный хостинг для Hadoop

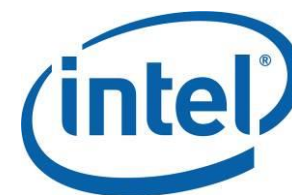
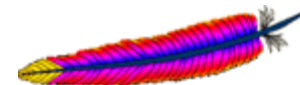
Экосистема Hadoop

- Pig – декларативный язык анализа данных
- Hive – анализ данных с использованием языка, близкого к SQL
- Oozie – поток работ в Hadoop
- Hbase – база данных (нереляционная), аналог Google Big Table
- Mahout – машинное обучение
- Sqoop – перенос данных из РСУБД в Hadoop и наоборот
- Flume – перенос логов в HDFS
- Zookeeper, MRUnit, Avro, Giraph, Ambari, Cassandra, HCatalog, Fuse-DFS и т.д.



Дистрибутивы Hadoop

- Apache
 - hadoop.apache.org
 - Оригинальный дистрибутив, только Hadoop
- Альтернативные дистрибутивы:
 - Совместно Hadoop, HBase, Pig, Hive, Mahout, Sqoop, Zookeeper и др.
 - Средства автоматизации установки и администрирования, мониторинг, безопасность
- Поставщики альтернативных дистрибутивов:
 - Cloudera
 - MapR
 - Hortonworks
 - Intel



Облачный хостинг Hadoop

- Amazon Elastic MapReduce (Amazon EMR)

- <http://aws.amazon.com/elasticmapreduce/>
- Партнерство с MapR



- Apache Hadoop on Rackspace

- http://www.rackspace.com/knowledge_center/article/apache-hadoop-on-rackspace-private-cloud
- Партнерство с Hortonworks



- Microsoft Windows Azure

- <http://www.windowsazure.com/en-us/home/scenarios/big-data/>



- Qubole Data Service

- <http://www.qubole.com/qubole-data-service>
- Web-интерфейс для анализа данных с Hadoop, Hive, Pig и др. на Amazon EMR



Ограничения Hadoop и другие модели

- Ограничения Hadoop
 - Только пакетная обработка (batch processing)
 - После обработки MapReduce данные записываются на диски
 - Только один алгоритм - MapReduce
- Google больше не использует MapReduce
 - Объявлено в 2014 г (<https://clck.ru/9Ts9u>)
 - Новая система Cloud Dataflow
- Обработка «потоковых» данных (streaming processing)
 - Apache Storm
- Обработка данных в памяти
 - Apache Spark

Вопросы?